Class Diagram Description Table

By: Sohail Mohammed, Elias Amal, Omar Khan, Tony Wu

Class	Method	Description
ErrorLogs	+ determineErrorMessage(error_type: string): String	Selects and creates an appropriate error message based on type of error (violations of constraint or file errors). It returns the error message in string format ERROR: <msg>
	+ passTicketConstraint(numTickects: int): boolean	Takes in the number of tickets left over from sell buy transaction recorded in Available Tickets File. Checks whether the number of tickets for an event is negative. If constraint is passed returns true, or else returns false.
	+ passUserConstraint(username: string): boolean	Takes as input the username and checks whether this username violates constraint that username should not be an existing user. Returns whether username passes constraint, true or false, fails constraint
ProcessCurrentUsers	+ getUsernameList(): ArrayList<String>	Reads the old current users file, finds new created users then stores the usernames into an ArrayList .
	findUserType(username: String): String	Takes in username of user as input parameter. Uses the username to search for the user type of the user in the current user file. Returns the user type as a string

	+ findUserAmount(username: String): float	Takes in username of user to search of the user account amount. The function returns the user amount as a float value.
ProcessAvailableTickets	+ ParseTickets(): void	
	+ getTickets(): ArrayList<Ticket>	Returns the and instance of tickets. Ticket instance contains information about tickets eg. number of tickets, price of ticket
UpdateUserAccounts	+ updateAmount(old_user_amount: List<float>, change_user_amount: List<float> , username: String)	Takes in the old user amount and the any changes in user amount lists, then merges the arrays based on the username
	dailyTransactionParse(): void	Parses the Dialy Transaction file to determining which format to use to parse line based on transaction code
	parseFormat1(line: String): void	Parse daily transaction file in the first format retrieving username, user type, credit amount
	parseFormat2(line: String): void	Parse daily transaction file in the second format retrieving the buyer's username, seller's username and the credit amount
	parseForamat3(line: String): void	Parses the daily transaction file in third format retrieving the event name, seller's username, number of tickets, and price of tickets.
	writeUsers(username_list: ArrayList<String>): void	Writes to the Current User Accounts files the updated values after accounting for

		changes made in daily transaction file
	writeTickets(event_title: List<String>): void	Writes to the Available Tickets files the updated values after accounting for changes made in daily transaction file